Earth Data discovery OpenSearch Best Practices

Earth Data discovery from a historical perspective

Earth science data abound in cyberspace, yet many of them are so-called "dark data", i.e., difficult or impossible to find in a meaningful way. Even data that are offered by well-funded science data archives may be accessible only through proprietary search interfaces. This complicates the life of a scientist trying to collect datasets from a variety of sources. Often, the job devolves to:

1. Discover that Dataset A is available at website X
2. Learn the user interface for website X
3. Conduct the search at website X
4. Order and then acquire data for website X through interface
5. Repeat 1-4 for Dataset B at website Y

Over time, several protocols have developed to support "one-stop-shopping". For example, the Z39.50 became popular in the 1990's for searching documents and still has some currency in the digital library community, though this is lacking in some of the spatial-temporal concepts needed for effective Earth Data science queries. Also in the 1990's was the EOSDIS Version 0 protocol, a standard developed at NASA to support dataset and granule level searching over sockets, and later over Hypertext Transfer Protocol (HTTP). More recently, the Catalog Services for the Web (CSW) protocol has been developed within the Open Geospatial Consortium (OGC), with many adherents particularly in the area of structured searches. However, both the EOSDIS V0 and CSW protocols can be complicated to implement (and even understand) for a science researcher with little or no information technology staff.

Abstract

The Discovery Cluster within the Earth Science Information Partners (ESIP) Federation and CEOS WGISS Integrated Catalog (CWIC) have developed a set of conventions around the ATOM syndication specification and OpenSearch 'query-response' specification to enable lightweight mechanisms of data discovery and access. The result is a data discovery framework that is,

- Lightweight and simple
- Standards-based
- REST-likeful
- Low entry cost

The purpose of this document is to achieve the following with respect to any Earth Data OpenSearch implementation,

- Promote the use of the OpenSearch standard as a means of data discovery for Earth Data providers
- Define the expectations and requirements of candidate OpenSearch implementations
- Remove ambiguity in implementation where possible
- Facilitate the aggregation of results between disparate Earth Data providers via OpenSearch common standards

- Allow for clients to access search engines via an Open Search Descriptor Document (OSDD) with no a priori knowledge of the interface
- Facilitate smooth integration between related OpenSearch implementations, such as a dataset resource collection that refers to granule resource collections from another provider (IDN and CWIC is a good example)

Table of contents

Introduction

The A9 search technology company, a subsidiary of Amazon, originally developed the OpenSearch specification, which was released to the general community in 2005, and adopted in a variety of settings. The OpenSearch convention is currently maintained on www.opensearch.org, a site provided by A9 to the community.

An Open Search Descriptor Document (OSDD) served by an OpenSearch implementation defines a discovery service in terms of,

- HTTP GET query URL, defining simple keyword/value pair query parameters
- Result formats it supports (ATOM, RSS, HTML)

Given that document, a client can execute a query via a simple HTTP GET invocation. The server will respond with a navigable result set containing references to pertinent inventory in terms of

- Spatial extent
- Temporal extent
- Metadata, data and documentation URLs
- Links to further searches (possibly represented by additional OSDDs)

These results can be used by clients that are as simple as a web browser or a complex as an aggregated portal implementation.

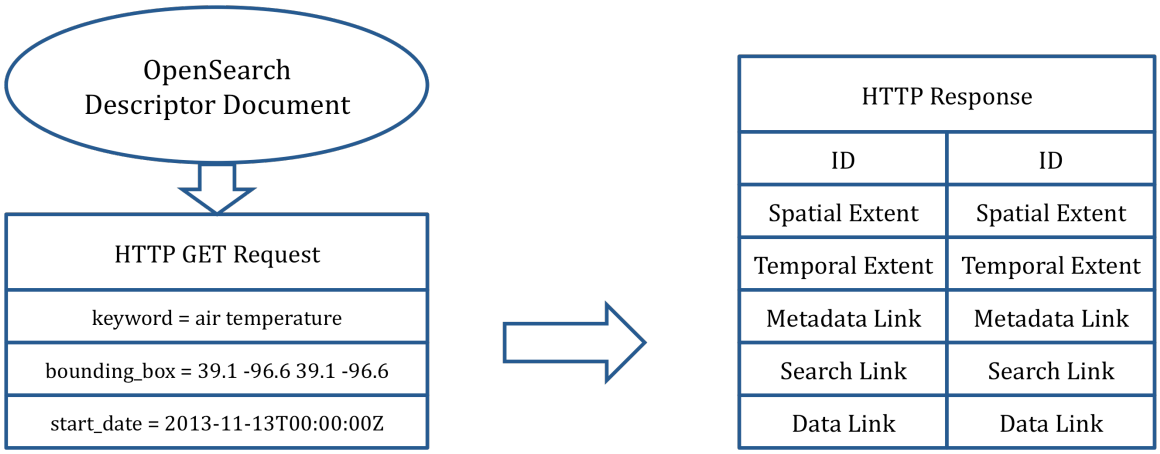

Figure 1: Search Architecture

Specification and extension adherence

Our best practices regard the following specifications and extensions as mandatory when implementing OpenSearch,

- ATOM Syndication format
- OpenSearch specification version 1.1 draft 5
- OpenSearch geo extension version 1.0 draft 2

- OpenSearch time extension version 1.0 draft 1
- OpenSearch geo and time extension - OGC10-032r8

Our best practices regard the following extensions as optional but recommended when implementing OpenSearch,

- OpenSearch parameter extension version 1.0 draft 2
- OpenSearch relevancy extension version 1.0 draft 1

The Open Search Descriptor Document

The primary purpose of an Open Search Descriptor Document (OSDD) is to describe to a client (machine or human) how to search an inventory and what results to expect. Searching is defined in terms of protocol, endpoint and query parameters. Results are defined in terms of format.

Augmenting an OSDD with parameter elements as per Draft 2 of the parameter extension gives us the ability to

1. Increase the specificity of our search parameters.
2. Allow a client, in theory, to programmatically construct a user interface for an arbitrary OpenSearch implementation on the fly solely from an OSDD.

Both of these abilities are crucial to one of our goals, the ability to aggregate discovery across multiple, disparate providers.

Anatomy of an OSDD

They key element of an OSDD is the Url element. Taken as a whole, including any child parameter elements, the entirety of an OpenSearch implementation for a collection of resources can be described to a client.

For example,

```
<Url type="application/atom+xml"
template="http://foo.gov/opensearch/datasets.atom?q={os:searchTerms?}”/
>
```

Tells a client that an inventory of ‘dataset’ resources can be searched from the endpoint ‘foo.gov/opensearch’ using the http protocol and that the results can be filtered by the optional query parameter ‘q’.
Binding the parameter ‘q’ to the ‘os:searchTerms’ descriptor conveys meaning. In this case, the OpenSearch specification defines this meaning as *‘keyword or keywords desired by the search client’*

Placeholders for optional search parameters in URL templates must include a question mark character ‘?’ at the end of the search term. However, the search-specific attributes themselves are not considered themselves to be part of the specification. Also, such attributes MUST be optional; required search-specific terms that are not spelled out in the OpenSearch specification render the associated search service non-compliant with that convention.

The type attribute defines the format of results returned by any query against this resource collection. In this case, the results format is ATOM.

Given the example above the following requests are valid,

```
http://foo.gov/opensearch/datasets.atom

http://foo.gov/opensearch/datasets.atom?q=foo
```

Given this example,

```
<Url type="application/atom+xml"
template="http://foo.gov/opensearch/datasets.atom?q={os:searchTerms}”/>
```

The following request is NOT valid,

http://foo.gov/opensearch/datasets.atom

because the required parameter “q” and attendant keywords are not included.

We require that an OpenSearch implementation supports the ATOM results format and that it may be requested via HTTP content negotiation or resource extension.

We can increase the specificity of a query parameter by leveraging the parameter extension. For example, an OpenSearch API that can only return a maximum of 2000 results can be described as follows,

```
<Parameter name="count" value="{os:count}" minimum="0"
maxInclusive="2000"/>
```

We recommend that an OpenSearch implementation use ‘parameter’ elements as per the OpenSearch parameter extension to describe their search-engine-specific search parameters.

A client may wish to generate a user interface at runtime by parsing the OSDD. Leveraging the parameter extension comes close[1] to allowing us to achieve this.

For example, the following parameter definition may instruct a UI to provide a temporal widget by virtue of the value ‘{time:start}’ which describes an RFC-3339 date time. That widget would have a display name of ‘Temporal Start’ and will be deemed optional by virtue of the minimum value of zero.

```
<Parameter name="startTime"
      uiDisplay= “Temporal Start”
```

[1] We will petition OpenSearch to add the ‘uiDisplay’ attribute to the parameter element

```
      value="{time:start}"
      title="inventory which has a temporal extent containing this start
time"
      minimum="0"/>
```

We require that an OpenSearch implementation use name, value, title and uiDisplay[1] attributes for their non-standard[2] parameter definitions.

Some of our common search parameters have characteristics that are difficult to capture using the parameter extension as it is in Draft 2.

Free text searching, for example, may support query syntax. Given that there is no widely adopted free text query syntax standard and that most implementations will provide this support through 3rd party products like Lucene or Elastic Search, we are proposing an augmentation to the parameter extension that will allow a parameter to exhibit a profile analogous to 'my free text search query language behaves like Lucene'

For example,

```
<Parameter name="keyword"
      value="{os:searchTerms}"
      title="inventory containing all the specified keywords separated
by space, case-insensitive, wildcards are supported"
      minimum="0">
      <link rel="profile"
      href="http://www.elasticsearch.org/guide/en/elasticsearch/referen
      ce/current/query-dsl-query-string-query.html"
      title="This parameter follows the elastic search free text search
      implementations" />
</Parameter>
```

We recommend that any OpenSearch implementation with free text search capabilities define the properties of that capability via an ATOM link referring to a profile.

The geometry parameter as defined in the geo extension allows for the representation of a spatial constraint via Well Known Text (WKT). This WKT string may represent a number of different geometries such as point, line or polygon. It is likely that an OpenSearch implementation only supports a subset of these geometries. We are proposing an augmentation to the parameter extension that will allow a parameter to exhibit a number of profiles analogous to 'my geometry search parameters supports point, line and polygon'

For example,

```
<params:Parameter name="geometry"
```

[2] By non-standard we mean parameters other than those defined in the OpenSearch specifications and extensions

```
      value="{geo:geometry}"
      title="inventory which has a spatial extent overlapping this
      geometry"
      minimum="0">
      <atom:link rel="profile"
      href="http://www.opengis.net/wkt/LINESTRING"
      title="This service accepts WKT LineStrings"/>
      <atom:link rel="profile"
      href="http://www.opengis.net/wkt/POINT"
      title="This service accepts WKT Points"/>
      <atom:link rel="profile"
      href="http://www.opengis.net/wkt/POLYGON"
      title="This service accepts WKT Polygons"/>
</params:Parameter>
```

We recommend that any OpenSearch implementation with geometry search capabilities define an enumeration of supported geometry types via ATOM links referring to geometry type profiles.

Obtaining an OpenSearch Descriptor Document

You may wish to track client usage through OpenSearch implementations for metrics purposes. You can do this by passing a client ID from implementation to implementation.

We recommend that an OpenSearch implementation expose its OSDD via the provision of a client-supplied identifier

In the search section of this document we talk about metrics related to an open search implementation. A key element of metrics support is the ability to identify a client and their usage of your API in some manner. To facilitate this we have introduced the concept of a non-mandatory client id parameter.

In order to minimize the obtrusiveness on the client of a client id we suggest the following,

1. Expose your OSDD via an OpenSearch html landing page.
2. Your OSDD is dynamically generated based on the submission of an html form on that page that has a single, mandatory 'clientId' parameter.
3. Submitting the form will return an OSDD with that client ID embedded in the URL template attribute.

For example invoking,

```
HTTP GET foo.gov/datasets/openSearchDescriptorDocument.xml?clientId=foo
```

Would return an OSDD containing,

```
<Url type="application/atom+xml"
template="http://foo.gov/opensearch/datasets.html?clientId=foo”/>
```

Consequently, if a client abides by the contract of the url template, all queries to your API will contain the clientId parameter.

The Search Request

An OpenSearch request is a HTTP request (normally 'GET' but this can be defined within your OSDD) directed to a specific collection of resources. Those resources normally describe datasets or granules in earth data discovery.

Anatomy of a Search Request

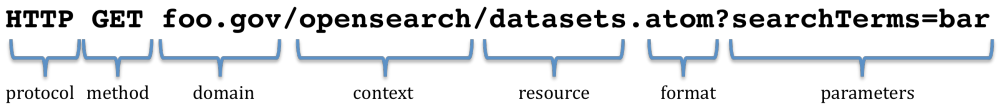

Figure 2: OpenSearch query URL

The set of resources returned can be constrained and traversed by attaching parameters to your request.
The requested format of your results can be defined in the request by content negotiation via the HTTP 'accept' header or using a resource extension.

Example of format request by resource extension,

```
https://foo.gov/opensearch/datasets.atom
```

Example of format request by HTTP 'accept' header,

```
Accept: application/xml+atom: https://foo.gov/opensearch/datasets
```

Query Parameters

We recommend the use of keyword, spatial and temporal constraints for your search implementation. You may define other search parameters but the above 3 have been found to account for over 90% of user queries in a recent study conducted by NASA's ECHO system.

It is expected that multiple query parameters will be AND'd together to form a query.

Search Terms – OpenSearch Specification

The searchTerms parameter is expected to be a simple set of keywords to be used in a free text search. This contrasts somewhat with the traditional method of searching (e.g., within EOSDIS Version 0), which emphasized structured searches based on specific attributes like satellite and instrument. However, free text search is simpler to implement at the client end and more universal, not relying on a

common schema, and with judicious use of acronyms by the client or user (e.g., "MODIS") can be nearly as precise. Note that searchTerms are an important discriminator for data collections; however, they are often not useful for files within a data collection, where the discriminator is more likely to be space or time coordinates (see 'two-step searching) of the files. Thus searchTerms is not a required parameter; rather it is the server that specifies in a template whether they are required for that particular search type.

Keyword constraints are key to discovering the appropriate data collections for a user.

Note that if your keyword search algorithm does not conform to any profile (Lucene or Elastic Search for example) then the following is expected:

When multiple searchTerms are included, they should be separated by '+', e.g., "MODIS+fires". Implementations are required to treat this combination as a Boolean "AND" operation by default. You should specify multi-word phrases by setting double quotes (") around the phrase. In this case, an implementation must treat this as a phrase search.

We require that an OpenSearch implementation, if appropriate, should filter results by a free text keyword constraint as specified in the OpenSearch specification

Spatial and temporal constraints narrow down a result set to a user's desired area and time of interest.

Spatial – OpenSearch Geo Extension

Bounding box is the simplest and most used type of spatial constraint. As per the geo extension we recommend search providers define a bounding box constraint as 4 longitude/latitude coordinates expressed as EPSG:4326 decimal degrees. The order of those coordinates should be west, south, east and north.

For example,

```
datasets.atom?box=-180.0,-90.0,180.0,90.0
```

The implied relation between a spatial constraint and any inventory returned is that the spatial extent of a resource overlaps the spatial constraint in the query.

Search providers may implement other spatial constraints specified in OpenSearch geo extension version 1.0 draft 2 but support for Geo Bounding Box support is required.

We require that an OpenSearch implementation should filter results by a geo bounding box constraint as specified in the OpenSearch Geo extension

Temporal – OpenSearch Time Extension

Temporal constraints can be expressed as intervals in time that may be open ended. A upper or lower temporal bound can be expressed as a date or date time constraint specified in the RFC-3339 format as per the OpenSearch Time extension.

Example of a fully specified range,

```
datasets.atom?startTime=2001-01-01T22:00:00Z&endTime=2001-01-
01T22:00:00Z
```

Example of an open-ended range,

```
datasets.atom?startTime=2001-01-01T22:00:00Z
```

Example of an open-ended range with date only,

```
datasets.atom?startTime=2001-01-01
```

The implied relation between a temporal constraint and any inventory returned is that the temporal extent of a resource overlaps the temporal constraint in the query.

We recommend that an OpenSearch implementation should filter results by time start and time end as specified in the OpenSearch Time extension

Result set navigation parameters

We recommend using paging to navigate through results sets. Any defaults used by your implementation when not supplied by the client should be rendered in your result.

We recommend that a startPage value of '1' refer to the first page of a result set.

Example,

```
datasets.atom?startPage=1&count=10
```

Will return the results 1 through 10 of a result set.

```
datasets.atom?startPage=2&count=10
```

Will return the results 11 through 20 of a result set.

Note that certain implementations may prefer to use the 'startIndex' parameter rather than 'startPage'. This is acceptable but not recommended.

We recommend that an OpenSearch implementation must allow a client to navigate a result set using the 'startPage' and 'count' parameters as specified in the OpenSearch specification

Metrics parameters

In the OSDD section we talk about the need for the user to supply a client ID when obtaining an OSDD. The OSDD returned will embed that client ID in the URL template. In theory, all requests sent by that client will have that parameter present.

The purpose of this ID is to allow the tracking of metrics based on particular clients and users.

To facilitate this need we recommend that search providers implementing OpenSearch use this client ID reference when logging or generating their own metrics. Furthermore, we recommend that the client ID be propagated to other searches (see 'Two step searching). We strongly recommend that the client ID should be a mandatory parameter when obtaining an OSDD through your API but optional in any search request.

We recommend that an OpenSearch implementation must allow a client to specify a 'clientId' and that ID must be propagated to any secondary OpenSearch implementations

While this is not an ideal means of providing user , it appears to be the best way to provide 'opt in', non-obtrusive support.
The supplied client ID may be (a) non-unique and (b) not meaningful. This risk is balanced by the possible lack of interest in a provider's OpenSearch API if a more thorough handshake (designated IDs for example) is required.

Example,

```
datasets.atom?clientId=foo
```

Full Example of a 'best practices' compliant Open Search Request

```
https://foo.gov/opensearch/datasets.atom?clientId=foo&searchTerms=air+t
emperature&box=10,10,10,10&startTime=startTime=2001-01-
01T22:00:00Z&endTime=2001-01-01T22:00:00Z&pageNumber=1&count=10
```

The Search Response

An OpenSearch response is an ATOM feed with zero or more ATOM entries. Each entry represents a single resource pertaining to the query submitted.

Anatomy of a response

An ATOM response is one ATOM feed element containing the following,

1. Information about the search implementation in terms of title, author and ID.
2. Information about the nature of the result set in terms of total number of results, number of results returned and how many the client asked for.
3. Navigation information for traversing that result set including links to the previous, next, first and last results in the set.
4. Zero or more entries pertaining to resources matching the client query

HTTP Response				
ATOM Feed				
Atom Entry 1	Atom Entry 2	Atom Entry 3	Atom Entry 4	Atom Entry 5
Metadata Link	Metadata Link	Metadata Link	Metadata Link	Metadata Link
Search Link	Search Link	Search Link	Search Link	Search Link
Data Link	Data Link	Data Link	Data Link	Data Link
Documentation Link	Documentation Link	Documentation Link	Documentation Link	Documentation Link

Figure 3: OpenSearch Response Architecture

Result set navigation by content

As described in the request section,

We recommend that an OpenSearch implementation must allow a client to navigate a result set using the 'startPage' and 'count' parameters as specified in the OpenSearch specification

This recommendation has an impact on how a provider describes their result set. For example,

```
<feed>
        …
        <os:totalResults>3415</os:totalResults>
        <os:itemsPerPage>10</os:itemsPerPage>
        <os:startPage>1</os:startPage>
        …
</feed>
```

Indicates that the client asked for the first page of results where there are 10 results per page and there are a total of 3415 results. The first 10 entries (1-10) would be represented in the feed. If the client asked for the second page (startPage=2) then the second 10 entries (11-20) would be represented in the feed.

In the case where zero results are returned the following would be contained in the feed,

```
<feed>
```

```
        …
        <os:totalResults>0</os:totalResults
        …
        <subtitle type="text">Your search yielded zero
matches</subtitle>

</feed>
```

An OpenSearch implementation should not include 'itemsPerPage' and 'startPage' in a feed result when zero entries are returned for a query and that the feed subtitle should contain human-readable text describing that state.

Result set navigation by HATEOAS

In order to facilitate traversal of a result set we can leverage the REST concept of Hypermedia as the Engine of Application State (HATEOAS) and provide links to the current, previous, next, first and last pages in the result set as follows,

```
<feed>
        …
        <os:totalResults>50</os:totalResults>
        <os:itemsPerPage>10</os:itemsPerPage>
        <os:startPage>3</os:startPage>
        <link
href="foo.gov/opensearch/datasets.atom?count=1&numberOfResults=10"
rel="first" type="application/atom+xml"/>
        <link
href="foo.gov/opensearch/datasets.atom?count=2&numberOfResults=10"
rel="prev" type="application/atom+xml"/>
        <link
href="foo.gov/opensearch/datasets.atom?count=3&numberOfResults=10"
rel="self" type="application/atom+xml"/>
        <link
href="foo.gov/opensearch/datasets.atom?count=4&numberOfResults=10"
rel="next" type="application/atom+xml"/>
        <link
href="foo.gov/opensearch/datasets.atom?count=5&numberOfResults=10"
rel="last" type="application/atom+xml"/>
        …
</feed>
```

We recommend that an OpenSearch implementation provides navigation links for the first, previous, current, next and last pages of a result set.

Result metadata

Each resource in a result will be represented by an ATOM 'entry' element. That element will contain an ID, a link to the metadata from which this entry was derived, and, if applicable, a spatial extent and a temporal extent.

An OpenSearch implementation should provide a link of relation 'alternate' for each entry in a result set feed. That link should point to the original metadata from which this entry was derived.

When rendering a spatial extent for an entry we recommend that an implementation provide a minimum-bounding rectangle (MBR) to represent more complex geometries such as polygon in addition to the accurate spatial extent. The rationale behind this is that bounding rectangle is the lowest common denominator of spatial constraint/extent supported by our providers and clients. You may also render your original extent in WKT as per the geo extension.

For example, an MBR representing a point and it's original representation,

```
<georss:box>39.1 -96.6 39.1 -96.6</georss:box>
<geo:geometry>39.1 -96.</geo:geometry>
```

An OpenSearch implementation should render spatial extents using a minimum-bounding rectangle in the 'georss' format as well as the native representation of that extent.

When rendering a temporal extent, an implementation should use a Dublin core date element, which will support date and date-time ranges, open-ended date and date-time ranges and single dated and date-times.

For example a date-time range,

```
<dc:date xmlns:dc="http://purl.org/dc/elements/1.1/">1984-12-
25T00:00:00.000Z/1988-03-04T00:00:00.000Z</dc:date>
```

an open-ended date-time range,

```
<dc:date xmlns:dc="http://purl.org/dc/elements/1.1/">1984-12-
25T00:00:00.000Z/</dc:date>
```

a single date-time,

```
<dc:date xmlns:dc="http://purl.org/dc/elements/1.1/">1984-12-
25T00:00:00.000Z</dc:date>
```

a single date,

```
<dc:date xmlns:dc="http://purl.org/dc/elements/1.1/">1984-12-
25</dc:date>
```

An OpenSearch implementation should render temporal extents using a Dublin Core date element.

Entries should also render link elements pointing to various artifacts associated with the resource. Those may include data, additional metadata, browse imagery and documentation.

An OpenSearch implementation should use the following relation values when describing artifacts associated with a resource,

Artifact	Definition	Relation	Example
Metadata	file with (usually) structured information about corresponding data files	via	<link href="foo.xml" rel="via"/>
Browse	image of the data typically used for making data request decisions	icon	<link href="sample.gif" rel="icon"/>
Documentation	File with human-readable information about the resource	describedBy	<link href="foo.pdf" rel="describedBy"/>
Data	link representing a data file or other science data resource; may be large in size	enclosure	<link href="foo.hdf" rel="enclosure"/>
OSDD	link to an OpenSearch Description Document; useful for recursive searching	search	<link href=”osdd.xml” rel=”search”/>

Table 1: Link relations for OpenSearch artifacts

Result ordering

Result ordering is optional as per OGC recommendations. We recommend that if you support search by free text that you employ relevancy ranking as per the Open Search Relevancy Extension and order your results by that ranking in descending order.

For example,

```
<feed xmlns:relevance="http://a9.com/-
/opensearch/extensions/relevance/1.0/"
>
        <entry>
                …
                <relevance:score>0.98</relevance:score>
                ..
        </entry>
        <entry>
                …
                <relevance:score>0.75</relevance:score>
                ..
        </entry>
        …
</feed>
```

Handling errors

In the event of an error the service implementation should respond with the appropriate HTTP error code and a response body that verbosely describes the error in ATOM format.
These errors will generally fall into two categories,

- User input errors will be responded to with a BAD_REQUEST 400 response or another error in the 4xx range, if the error corresponds to one of the existing HTTP error codes.
- Processing errors will be responded to with an INTERNAL_SERVER_ERROR 500 response or another error in the 5xx range, if the error corresponds to one of the existing HTTP error codes.

Note that a query that yields zero results is not treated as an error state and, therefore, should return a successful HTTP code.

Two-step Searching

One serious hurdle to overcome in searching for data is the enormous number of data items to account for in responses, as well as the expected number of successful "hits" for a query. In ordinary web searches, the searcher is usually looking for a small number of web pages or documents. Relevance ranking typically does a good job of presenting these successful hits near the top of the returned list, followed by single point-and-click retrievals. However, when searching for Earth science data covering large time periods or spatial areas, a user will often specify a set of constraints to find an appropriate data collection together with space-time criteria for files within that data collection. Often, the precision of the data collections returned for the search is low, with many spurious hits. However, the space-time precision of the files is often quite high: that is, the user truly wants to use all the data files of a desirable data collection set that fall within the space-time region of interest. Thus, searching for all data satisfying both dataset content and space-time region at the same time can produce a great many spurious hits, i.e., all the files for data collections that are *not* desired.

The two-step solution breaks a search into two distinct parts. A client will first identify datasets of interest (step 1) and then search for granules associated with those datasets (step 2).

To facilitate this, when an OpenSearch implementation representing dataset/collection resources returns results each entry in that result set should contain an ATOM link of relation type 'search' if applicable. That ATOM link should point to a URL of an OSDD pertaining to granules associated with that entry (i.e. Dataset/collection). A client can then parse that OSDD and formulate a search for granules belonging to that dataset.

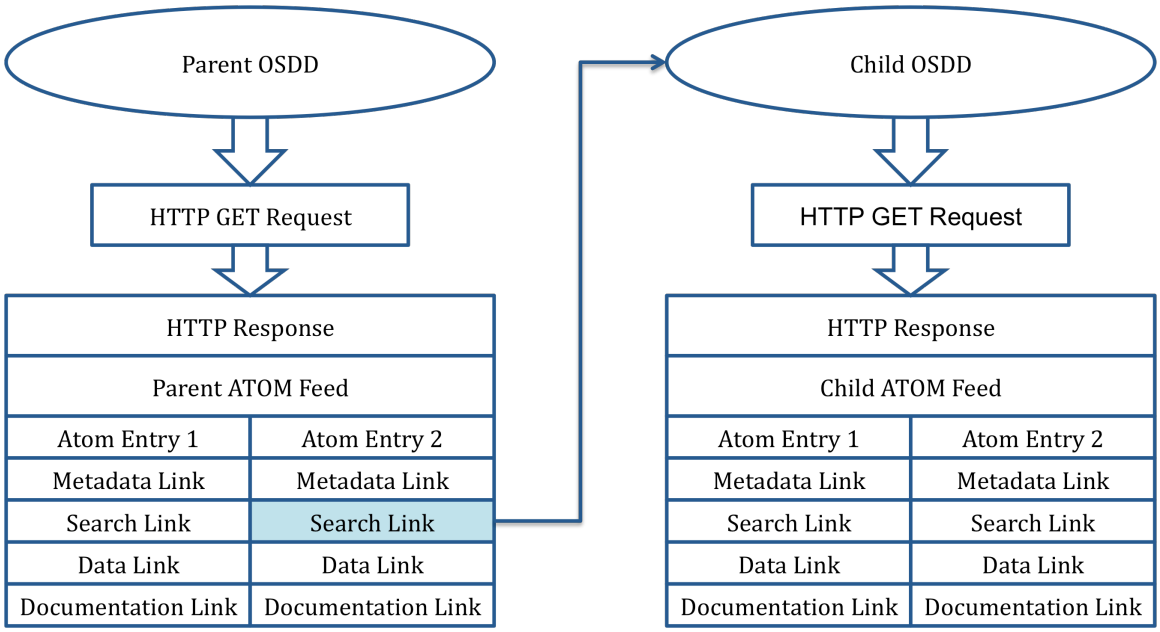

Figure 4: Two-step architecture

For example,

1. Given an dataset OSDD, search for datasets of interest

```
GET http://foo.gov/opensearch/datasets?searchTerms=MODIS&clientId=d
```

2. Parse search results

```
<feed … >
   …
   <entry … >
    <id>http://foo.gov/opensearch/datasets/MODIS_dataset</id>
         <link rel='search'
type='application/opensearchdescription+xml'>bar.gov/opensearch/gran
ules/descriptorDocument?datasetId=MODIS_dataset&clientId=d</link>
         …
   </entry>
   …
</feed>
```

3. Retrieve granule OSDD
4. Parse granule OSDD

```
<os: OpenSearchDescription>
     …
     <os:Url type="application/atom+xml"
         template="bar.gov/opensearch/granules.atom?datasetId=
MODIS_dataset&clientId=d&box={georss:box?}">
               …
     </os:Url>
     …
```

```
</os:OpenSearchDescription>
```

5. Search for granules of interest

```
GET http://bar.gov/opensearch/granules.atom?datasetId=
MODIS_dataset&clientId=d&box=1,2,3,4
```

Although two steps, dataset and granule, are illustrated here, the notion of more steps could be used. That is, one could have an OpenSearch server that searches top-level (search engine) documents, returning the URLs to OSDDs for each search engine.
Each step in the process returns links to OpenSearch Description documents, which in turn inform the client on how to execute the next level search, until the lowest level, where the return consists of links to actual data of interest.

The two-step search concept is an optional aspect of the convention. There are use cases where only a dataset query is needed, as well as some where only the granule-level query is needed.

Versioning

In many cases, a client must know what version of our conventions an OpenSearch implementation is conforming to. Features may be added (e.g., new elements in the response) or in some cases even rolled back in subsequent versions. Thus, clients need an easy way to identify which version a given document conforms to.

Declaring an ESIP namespace within the root element of the document, would identify the OpenSearch Best Practices to which an implementation complies.

The convention for the namespace prefix is "esipdiscovery". The namespace URI to which it points is

http://commons.esipfed.org/ns/discovery/<version>/

The version must also be included as

esipdiscovery:version="<version>"

Thus, the opening element for an ESIP-compliant ATOM feed may look like:

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
      xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/"
      esipdiscovery:version="1.2">
```

Likewise, the beginning of an OpenSearch Description Document would look like:

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/"
    xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/”
    esipdiscovery:version=”1.2”>
```

The version that this Best Practices document pertains to is Version 1.2 of the ESIP Discovery specification.

Appendix A – Summary Table

Element or Attribute	Description	Namespace	Example
version *attribute*	Version of the ESIP Discovery Atom-based Specification supported by a search or collection casting provider. Required in OSDD and Atom response.	esipdiscovery	<?xml version="1.0" encoding="UTF-8"?> <OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/" xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/” **esipdiscovery:version=”1.2”**>
title element	Human-readable text describing about an entry, usually relatively short	ATOM	
id element	Unique identifier for a returned resource in ATOM response	ATOM	
link element	URL for a resource referenced in ATOM response	ATOM	
rel attribute	Relationship represented by a given <link> element.	ATOM	rel=”enclosure” (used to describe <link> elements pointing to data resources)
type attribute	Type of resource, typically given as a mime-type	ATOM	type=”application/x-netcdf” (describes link elements that are netCDF files).
Url	External resource location	OpenSearch	<Url type="text/html" template="http://example.com/search?q={searchTerms}&pw={startPage?}" />

template attribute	template attribute of <Url> element Required in OSDD	OpenSearch	
date	Temporal coverage of data in an ATOM response.	Dublin Core	
geo:box	Minimum bounding rectangle, required in ATOM response if spatial information is included	OpenSearch Geo	
georss:geometry	Optional Well-Known-Text based geometry	GeoRSS	

Table 2: summary of common elements and attributes

Appendix B – Examples

OSDD with parameter extension

```
<?xml version="1.0"?>
<os:OpenSearchDescription
      xmlns:os="http://a9.com/-/spec/opensearch/1.1/"
      xmlns:echo="http://www.echo.nasa.gov/esip"
      xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
      xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"
      xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/
" esipdiscovery:version="1.2"
      xmlns:params="http://a9.com/-
/spec/opensearch/extensions/parameters/1.0/"
      xmlns:atom="http://www.w3.org/2005/Atom" >
      <os:ShortName>ECHO Dataset Search</os:ShortName>
      <os:Description>NASA ECHO Dataset search using geo, time and
parameter extensions</os:Description>
      <os:Tags>ECHO NASA CWIC CEOS ESIP OGC dataset</os:Tags>
      <os:Contact>support@echo.nasa.gov</os:Contact>
      <os:Url
            type="application/atom+xml"
            params:method="GET"
            template="https://api.echo.nasa.gov:443/opensearch/datasets
            .atom?keyword={os:searchTerms?}&amp;instrument={echo:instru
            ment?}&amp;satellite={echo:satellite?}&amp;boundingBox={geo
            :box?}&amp;geometry={geo:geometry?}&amp;placeName={geo:name
            ?}&amp;startTime={time:start?}&amp;endTime={time:end?}&amp;
            cursor={os:startPage?}&amp;numberOfResults={os:count?}&amp;
            uid={geo:uid?}&amp;clientId=foo">
            <params:Parameter
                  name="keyword"
                  value="{os:searchTerms}"
                  minimum="0">
                  <atom:link
```

```
            rel="profile"
            href="http://www.elasticsearch.org/guide/en/ela
            sticsearch/reference/current/query-dsl-query-
            string-query.html"
            title="This parameter follows the elastic
            search free text search implementations" />
</params:Parameter>
<params:Parameter
      name="instrument"
      value="{echo:instrument}"
      title="Inventory associated with a satellite
      instrument expressed by this short name"
      minimum="0"/>
<params:Parameter
      name="satellite"
      value="{echo:satellite}"
      title="Inventory associated with a Satellite/platform
      expressed by this short name" minimum="0"/>
<params:Parameter
      name="boundingBox"
      value="{geo:box}"
      title="Inventory with a spatial extent overlapping
      this bounding box"
      minimum="0"/>
<params:Parameter
      name="geometry"
      value="{geo:geometry}"
      title="Inventory with a spatial extent overlapping
      this geometry"
      minimum="0">
      <atom:link
            rel="profile"
            href="http://www.opengis.net/wkt/LINESTRING"
            title="This service accepts WKT LineStrings"/>
      <atom:link
            rel="profile"
            href="http://www.opengis.net/wkt/POINT"
            title="This service accepts WKT Points"/>
      <atom:link
            rel="profile"
            href="http://www.opengis.net/wkt/POLYGON"
            title="This service accepts WKT Polygons"/>
</params:Parameter>
<params:Parameter
      name="placeName"
      value="{geo:name}"
      title="Inventory with a spatial location described by
      this name" minimum="0"/>
<params:Parameter
      name="startTime"
      value="{time:start}"
      title="Inventory with a temporal extent containing
      this start time" minimum="0"/>
<params:Parameter
      name="endTime"
      value="{time:end}"
      title="Inventory with a temporal extent containing
      this end time" minimum="0"/>
```

```
            <params:Parameter
                  name="cursor"
                  value="{os:startPage}"
                  minimum="0"/>
            <params:Parameter
                  name="numberOfResults"
                  value="{os:count}"
                  minimum="0"
                  maxInclusive="2000"/>
            <params:Parameter
                  name="uid"
                  value="{geo:uid}"
                  title="Inventory associated with this unique ID"
                  minimum="0"/>
      </os:Url>
      <os:Query
            role="example"
            echo:instrument="AMSR-E"
            echo:satellite="Aqua"
            title="Sample search"
            geo:box="-180.0,-90.0,180.0,90.0"
            time:start="2002-05-04T00:00:00-0400"
            time:stop="2009-05-04T00:00:00-0400"/>
      <os:Attribution>NASA ECHO</os:Attribution>
      <os:SyndicationRight>open</os:SyndicationRight>
</os:OpenSearchDescription>
```

OSDD without parameter extension

```
<?xml version="1.0"?>
<os:OpenSearchDescription
      xmlns:os="http://a9.com/-/spec/opensearch/1.1/"
      xmlns:echo="http://www.echo.nasa.gov/esip"
      xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
      xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"
      xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/
" esipdiscovery:version="1.2"
      xmlns:params="http://a9.com/-
/spec/opensearch/extensions/parameters/1.0/"
      xmlns:atom="http://www.w3.org/2005/Atom" >
      <os:ShortName>ECHO Dataset Search</os:ShortName>
      <os:Description>NASA ECHO Dataset search using geo, time and
parameter extensions</os:Description>
      <os:Tags>ECHO NASA CWIC CEOS ESIP OGC dataset</os:Tags>
      <os:Contact>support@echo.nasa.gov</os:Contact>
      <os:Url
            type="application/atom+xml"
            params:method="GET"
            template="https://api.echo.nasa.gov:443/opensearch/datasets
            .atom?keyword={os:searchTerms?}&amp;instrument={echo:instru
            ment?}&amp;satellite={echo:satellite?}&amp;boundingBox={geo
            :box?}&amp;geometry={geo:geometry?}&amp;placeName={geo:name
            ?}&amp;startTime={time:start?}&amp;endTime={time:end?}&amp;
            cursor={os:startPage?}&amp;numberOfResults={os:count?}&amp;
            uid={geo:uid?}&amp;clientId=foo">
      </os:Url>
```

```
      <os:Query
            role="example"
            echo:instrument="AMSR-E"
            echo:satellite="Aqua"
            title="Sample search"
            geo:box="-180.0,-90.0,180.0,90.0"
            time:start="2002-05-04T00:00:00-0400"
            time:stop="2009-05-04T00:00:00-0400"/>
      <os:Attribution>NASA ECHO</os:Attribution>
      <os:SyndicationRight>open</os:SyndicationRight>
</os:OpenSearchDescription>
```

ATOM result

```
<?xml version="1.0" encoding="UTF-8"?>
<feed esipdiscovery:version="1.2"
      xmlns=”http://www.w3.org/2005/Atom”
      xmlns:dc="http://purl.org/dc/terms/"
      xmlns:echo="http://www.echo.nasa.gov/esip"
      xmlns:esipdiscovery="http://commons.esipfed.org/ns/discovery/1.2/
      "
      xmlns:georss="http://www.georss.org/georss/10"
      xmlns:gml="http://www.opengis.net/gml" xmlns:os="http://a9.com/-
      /spec/opensearch/1.1/" xmlns:time="http://a9.com/-
      /opensearch/extensions/time/1.0/">
      <updated>2014-08-01T17:22:04.288Z</updated>
      <id>https://api.echo.nasa.gov:443/opensearch/datasets.atom</id>
      <author>
            <name>ECHO</name>
            <email>support@echo.nasa.gov</email>
      </author>
      <title type="text">ECHO dataset metadata</title>
      <os:totalResults>3434</os:totalResults>
      <os:itemsPerPage>1</os:itemsPerPage>
      <os:startPage>1</os:startPage>
      <os:Query os:searchTerms="FIFE" role="request"/>
      <subtitle type="text">Search for ‘FIFE’</subtitle>
      <link
            href="https://api.echo.nasa.gov:443/opensearch/granules/des
            criptor_document.xml"
            hreflang="en-US"
            rel="search"
            type="application/opensearchdescription+xml"/>
      <link
            href="https://api.echo.nasa.gov/opensearch/datasets.atom?
            keyword=FIFE&numberOfResults=1&cursor=1"
            hreflang="en-US"
            rel="self"
            type="application/atom+xml"/>
      <link
            href="https://api.echo.nasa.gov/opensearch/datasets.atom?
            keyword=FIFE&numberOfResults=1&cursor=3434"
            hreflang="en-US"
            rel="last"
            type="application/atom+xml"/>
```

```
<link
      href="https://api.echo.nasa.gov/opensearch/datasets.atom?
      keyword=FIFE&numberOfResults=1&cursor=2"
      hreflang="en-US"
      rel="next"
      type="application/atom+xml"/>
<link href="https://api.echo.nasa.gov/opensearch/datasets.atom?
      keyword=FIFE&numberOfResults=1&cursor=2"
      hreflang="en-US"
      rel="first"
      type="application/atom+xml"/>
<link
      href="https://wiki.earthdata.nasa.gov/display/echo/Open+Sea
      rch+API+release+information"
      hreflang="en-US"
      rel="describedBy"
      title="Release Notes"
      type="text/html"/>
<entry>
      <id>https://api.echo.nasa.gov:443/opensearch/datasets.atom?
      uid=C179003030-ORNL_DAAC</id>
      <author>
            <name>ECHO</name>
            <email>support@echo.nasa.gov</email>
      </author>
      <title type="text">15 Minute Stream Flow Data: USGS
      (FIFE)</title>
      <summary type="text">ABSTRACT: USGS 15 minute stream flow
      data for Kings Creek on the Konza Prairie</summary>
      <updated>2008-12-02T00:00:00.000Z</updated>
      <echo:datasetId>15 Minute Stream Flow Data: USGS
      (FIFE)</echo:datasetId>
      <echo:shortName>doi:10.3334/ORNLDAAC/1</echo:shortName>
      <echo:versionId>1</echo:versionId>
      <echo:dataCenter>ORNL_DAAC</echo:dataCenter>
      <echo:processingLevelId>3</echo:processingLevelId>
      <link
            href="http://daac.ornl.gov/cgi-
            bin/dsviewer.pl?ds_id=1"
            hreflang="en-US"
            rel="enclosure"/>
      <link
            href="http://daac.ornl.gov/FIFE/guides/15_min_strm_fl
            ow.html"
            hreflang="en-US"
            rel="describedBy"
            title="USGS 15 minute stream flow data for Kings
            Creek on the Konza Prairie (VIEW RELATED
            INFORMATION)"/>
      <georss:point>39.1 -96.6</georss:point>
      <georss:box>39.1 -96.6 39.1 -96.6</georss:box>
      <link
            href=”http://gcmd.nasa.gov/getdif.htm?FIFE_STRM_15M”
            hreflang="en-US"
            rel="enclosure"
            title="doi:10.3334/ORNLDAAC/1" type="text/html"/>
      <link
```

```
                href="https://api.echo.nasa.gov:443/opensearch/granul
                es.atom?clientId=our_html_ui&amp;shortName=doi:10.333
                4/ORNLDAAC/1&amp;versionId=1&amp;dataCenter=ORNL_DAAC
                "
                hreflang="en-US"
                rel="search"
                title="Search for granules"
                type="application/atom+xml"/>
        <link
                href="https://api.echo.nasa.gov:443/opensearch/granul
                es/descriptor_document.xml?clientId=our_html_ui&amp;s
                hortName=doi:10.3334/ORNLDAAC/1&amp;versionId=1&amp;d
                ataCenter=ORNL_DAAC"
                hreflang="en-US"
                rel="search"
                title="Custom ECHO Granule Open Search Descriptor
                Document"
                type="application/opensearchdescription+xml"/>
        <link
                href="https://api.echo.nasa.gov:443/catalog-
                rest/echo_catalog/datasets/C179003030-ORNL_DAAC.xml"
                hreflang="en-US"
                rel="alternate"
                title="Product metadata"
                type="application/xml"/>
                <dc:date>1984-12-25T00:00:00.000Z/1988-03-
                04T00:00:00.000Z</dc:date>
    </entry>
</feed>
```